Graph Based Statistical Analysis of Network Traffic

Hristo Djidjev
Los Alamos National Lab
djidjev@lanl.gov

Gary Sandine
Los Alamos National Lab
gars@lanl.gov

Curtis Storlie
Los Alamos National Lab
storlie@lanl.gov

Scott Vander Wiel
Los Alamos National Lab
scottv@lanl.gov

ABSTRACT

We propose a method for analyzing traffic data in large computer networks such as big enterprise networks or the Internet. Our approach combines graph theoretical representation of the data and graph analysis with novel statistical methods for discovering pattern and time-related anomalies. We model the traffic as a graph and use temporal characteristics of the data in order to decompose it into subgraphs corresponding to individual sessions, whose characteristics are then analyzed using statistical methods. The goal of that analysis is to discover patterns in the network traffic data that might indicate intrusion activity or other malicious behavior.

Categories and Subject Descriptors

G.2.2 [**Graph Theory**]: Graph algorithms; I.5 [**Pattern Recognition**]: Models—*Statistical*; I.2.6 [**Learning**]: Parameter learning

General Terms

Algorithms, Security, Theory, Experimentation

Keywords

Networks, Protocol Graphs, Graph Decomposition, Patterns, Statistical Modeling, Anomaly Detection

1. INTRODUCTION

There are two main approaches for detecting malware and attacks in computer systems: signature based, where a large database of attack signatures is used to identify an on-going attack, and anomaly based, in which an unusual activity pattern is identified. The anomaly detection approach has the advantage that new types of attacks could be identified even before their signatures are discovered and catalogued. To this end, such systems analyze regular users' activity data and build a model of "normal" activity, which is then compared with real time activity data. Any anomaly, i.e., significant deviation from the normal patterns in historical data, will be considered as potentially triggered by a cyber attack and further investigated.

One issue associated with such an approach is that there is typically no information in the available data about which user generates each particular session and which sessions are generated by the same user at roughly the same time and, hence, define a usage pattern. Even information about which packets belong to the same sessions is only implicit in the available data. For that reason, other researchers have focused on global, summary characteristics of the graph representing the traffic, rather than analyzing the traffic of individual users. For instance, Iliofotou et al. [6] and [5] analyze graph characteristics like sizes of graph components and degree distributions. Collins [1] studies the total number of connected components and its change over time. Our goal goes beyond that, we would like to distinguish between interrelated connections and unrelated connections and we would like to merge dependent connections (only) into a graph-represented patterns. Having information about the traffic of individual users will help us to determine paths regularly employed by users in order to reach network resources of interest or compute statistics such as the likelihood that, say, there is a usage pattern for going from host a to host b and from host b visiting hosts c and d. Another type of information that will increase the accuracy of anomaly detection would be if one can combine the information about the traffic patterns with time information, i.e., how the traffic depends on the time of day or the time of week.

In this paper we propose a graph-based method for analyzing traffic patterns in a large computer network in combination with novel statistical methods for discovering time-related anomalies in data with diurnal trends. We model the traffic as a graph and extract the subgraphs corresponding to individual sessions and use them to develop a statistical model for the network traffic. The goal of our analysis is to discover patterns in the network traffic data that might indicate intrusion activity or other malicious behavior. We use Cisco NetFlow records collected at intranet and Internet boundaries within a large computer network to construct graphs describing the host-to-server connections in given time intervals.

Our methodology can be used to design an anomaly detector

MLG '11 San Diego, CA, USA

that consists of two interacting modules. The *training module* analyzes a database of previously recorded or streaming traffic data and produces a proper model of "normal" traffic patterns. The *detection module* analyzes the between-host traffic in real time, extracting traffic patterns, and matching them against the usage model in order to estimate how likely the new traffic is, given the historical data. For this end, for each pattern (subgraph) in the new traffic data, a number is associated indicating the likelihood that it corresponds to normal (legitimate) activity. The two modules described above process the data simultaneously, since, while the detection module checks any new traffic for anomalies, the training module updates the statistical model (usage statistics), as the time moves along. Clearly, our approach requires very fast algorithms for extracting and matching patterns in order to produce results in real time, so we need to address the algorithmic challenges of our methodology.

We test our framework on data collected at Los Alamos internal network. We use Cisco NetFlow records collected at intranet and Internet boundaries within a large computer network to construct graphs describing the host-to-server connections in given time intervals.

2. RELATED WORK

The value of analyzing the social relationships between hosts and servers and in detecting anomalies in the global structure of the graph representing the communications in a network has been studied in several works, e.g., [2, 5, 6, 12]. However, these methods tend to lack fine-grained locality, something that our approach specifically address.

In this way, our work has some similarity to scan statistics, which have been widely used to detect local clusters of events [3, 7, 10, 11]. The scan statistic approach relies on defining a specified set of sub-regions in space and/or time and looking for discrepancies in these sub-regions.

Star type regions around nodes have been used by [14] in a scan statistic approach. Here we construct regions of interest (telescoping subgraphs) in a more sophisticated and dynamic way in order to closely capture the actions of an attacker in the network.

Communications in a computer network are also treated in a local manner in [4]. However, in [4], the anomaly detection is conducted on each individual edge, so that they are not leveraging the strength of pooling a neighborhood of edges as we do in this paper.

3. PROTOCOL GRAPHS

We collect packet header data from routers and network taps in a large enterprise network. The captured packet headers are assembled into flows that describe activity between hosts in the network.

3.1 Secure shell (SSH) protocol flows

While the techniques of our methodology will, with minor modifications, work for any type of communication protocols, and in future we do plan to apply it to different protocols and to a combination of protocols, for the analysis in this paper we consider the subset of flows that we can identify as using the Secure Shell (SSH) protocol. The SSH protocol allows encrypted communication connections between hosts and servers and is typically used for interactive login to remote hosts or for copying files. The subset of SSH flows in our flow log data set is easy to identify since the target port for initializing SSH connections is the standard TCP port 22. It is possible to run SSH servers that accept connections on other ports, but for our analysis we only consider connections to the standard port.

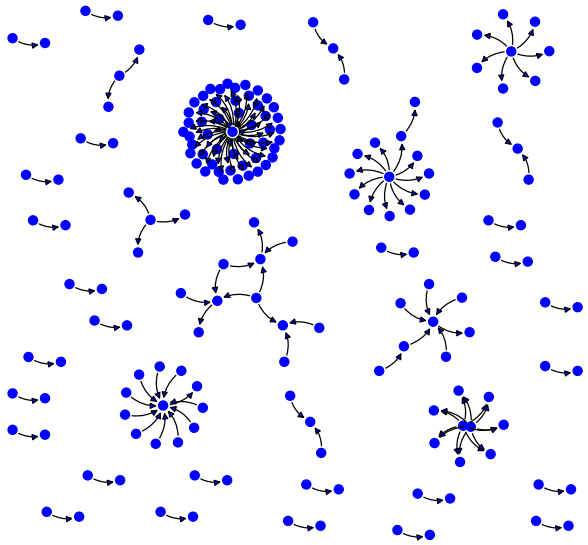

Figure 1: An SSH protocol graph for a 10 minute interval. The nodes are individual hosts in the network and the connections indicate a flow that occurred between hosts during the interval. Multiple (parallel) edges between hosts are not shown.

3.2 Construction of SSH protocol graphs

We represent the set of all SSH flow records by a directed multigraph $G(V, E)$, where the node set V is the set of all Internet Protocol (IP) addresses of the hosts and servers encountered in the data set and the directed edges E represent the observed SSH sessions starting at a client host and connecting to a server (at TCP port 22). Each edge e has time labels $s(e)$ and $f(e)$ that indicate the start and finish times of the flow. Note that G is a multigraph, which means that there might be multiple edges between some pairs (v, w) of nodes, corresponding to the case where multiple sessions occur between v and w at different times. We call G a *protocol graph* (Figure 1).

4. DECOMPOSITION ALGORITHMS

4.1 Telescoping graphs (TSGs)

Our objective is to partition a protocol graph G into subgraphs that we call telescoping subgraphs that are likely to correspond to a set of interrelated SSH sessions initiated by a single user or attacker. We call such set a *supersession*. For instance, in a supersession, a user can start at host a,

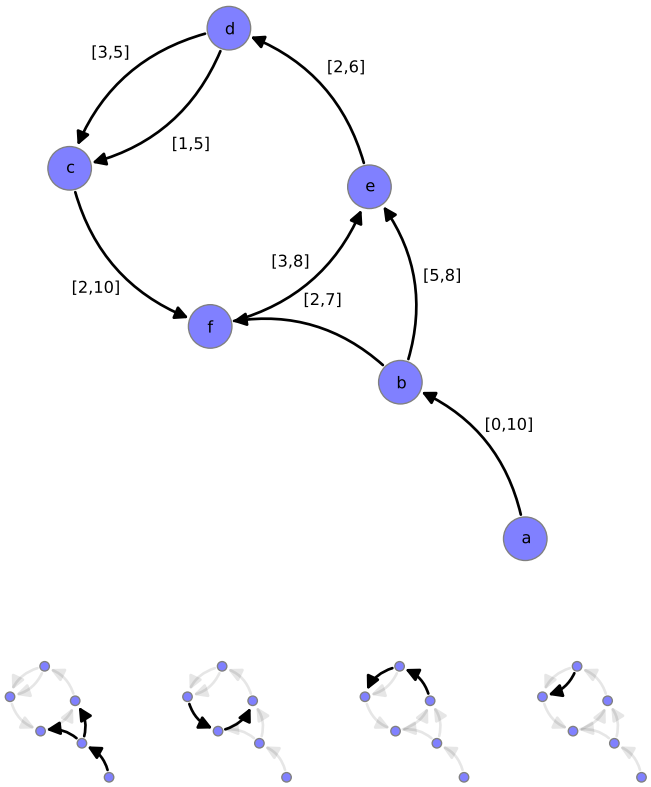

Figure 2: TSG decomposition. Pairs of numbers on each edge denote the start and the end times of the session. (a) A protocol graph G. (b) Decomposition of G into TSGs.

SSH from host a to b, go from b to c, and after finishes a job at c goes from b to d. Such a supersession will be represented by a TSG with a node set $\{a, b, c, d\}$ and with an edge set $\{(a, b), (b, c), (b, d)\}$. Our goal is to represent G as a union of TSGs and to design a very efficient algorithm for computing such a decomposition. First we formalize the notion of a TSG.

A *telescoping graph (TSG)* is a connected directed graph T that satisfies the following two conditions (Figure 2):

1. For any two edges $e_1 = (v, w)$ and $e_2 = (w, u)$ of T, the inequalities
$$s(v, w) \leq s(w, u) \text{ and } f(v, w) \geq f(w, u) \qquad (1)$$
are satisfied.
2. There is a node r called *root* with no incoming edges such that all nodes in T are reachable from r.

The inequalities (1), which we call the *telescoping conditions*, ensure that the session represented by edge e_1 should be active during the entire period when the session represented by edge e_2 is open, or otherwise the second session cannot be initiated by the first one. The root node r from Condition 2 corresponds to a host from which the entire supersessions is initiated.

If the starting and finishing times of the edges of G are distinct, then each TSG is a *directed acyclic graph (DAG)*, meaning that it does not have a directed path from a node to itself.

A *TSG-decomposition* of G is a sequence of TSGs $S = \{T_1, \ldots, T_k\}$ satisfying the following conditions (Figure 2):

1. Any two TSGs in S are edge-disjoint.
2. The union of all TSGs in S covers all nodes and edges of G.
3. For $i = 1, \ldots, k$, T_i is a *maximal* TSG in the graph $G_i = G \setminus (T_1 \cup \cdots \cup T_{i-1})$, meaning that there is no other TSG of G_i that contains T_i and has more edges.

It is clear that a TSG decomposition exists for any protocol graph, but it may not be unique. For instance, if the edge (b, f) from Figure 2 had a time label $[3, 11]$, the edge (f, e) could form TSG either with (b, f), forming a TSG consisting of the set of edges $\{(b, f), (f, e)\}$, or with (c, f), forming the TSG with edges $\{(c, f), (f, e)\}$. In order to deal with the ambiguity, we introduce two methods to break the ties:

(i) If (v, w) is an edge in G and $(u_1, v), \ldots (u_k, v)$ are all edges in G with target v that satisfy the telescoping condition, sorted in order of increasing start times, then (u_1, v) and (v, w) are in the same TSG. We call such decomposition an *earliest-first* TSG decomposition.

(ii) In the same situation as above, we choose the edge (u_k, v) that has latest start time to be in the same TSG as (v, w).

We call such decomposition a *latest-first* TSG decomposition. There are other possible strategies that can be used, but in this paper we don't study this issue in detail. Hereafter, if not specified otherwise, we assume that the decompositions are latest-first TSG decompositions.

4.2 Efficient computation of a TSG decomposition

Our algorithm for computing a latest-first TSG-decomposition of G first constructs an ancillary graph H associated with G. H has a node for each edge of G and an edge between any two nodes v_1 and v_2 of H whose corresponding edges e_1 and e_2 would be adjacent in a TSG of the desired decomposition, i.e., such that (i) e_1 and e_2 belong to the same TSG in a latest-first TSG-decomposition, and (ii) e_1 and e_2 are "adjacent", meaning that the target of e_1 is the source of e_2 or vice versa. Then the TSGs are exactly the connected

components of H and can be computed in time linear to the number of the edges of H.

In order to construct H, for each node v of G the algorithm considers the set S_{out} of all outgoing edges from v and the set S_{in} of all incoming edges in v. It then connects each edge e_{out} of S_{out} with the first edge e_{in} of S_{in} for which the pair of edges (e_{in}, e_{out}) satisfies the telescoping conditions. The order of the edges in S_{in} is essential. Since we are interested in a latest-first decomposition, the order of the incoming edges should be decreasing with respect to the starting time. For that reason, the edges of S_{in} are sorted in decreasing order prior to computation of the edges of H corresponding to node v..

Similarly, if the sorting of S_{in} is done in increasing order with respect to the staring time, the resulting modified algorithm will construct an earliest-first TSG decomposition.

The running time of the algorithm is dominated by the time for computing H. Let, for any node v in G, $\text{in}(v)$ and $\text{out}(v)$ denote the number of incoming and outgoing edges for v, respectively, and let $\deg(v) = \text{in}(v) + \text{out}(v)$. Let

$$h(v) = \frac{2}{1/\text{in}(v) + 1/\text{out}(v)}$$

denote the harmonic mean of $\text{in}(v)$ and $\text{out}(v)$ and let $h_{max} = \max_{v \in G} h(v)$. Then the time for that loop can be estimated by

$$T = O\left(|V(G)| + \sum_{v \in V(G)} (\text{in}(v)\text{out}(v))\right)$$

$$= O\left(n + \sum_{\substack{v \in V(G) \\ \deg(v) > 0}} \left(\frac{\text{in}(v)\,\text{out}(v)}{\text{in}(v) + \text{out}(v)}(\text{in}(v) + \text{out}(v))\right)\right)$$

$$= O\left(n + h_{max} \cdot \sum_{v \in G} \deg(v)\right) = O(n + h_{max} m),$$

where $n = |V(G)|$ and $m = |E(G)|$. The analysis shows that, for classes of graphs G for which $h_{max} = O(1)$, the running time of Algorithm 1 is linear in the size (the number of the nodes plus the number of the edges) of G. In our data h_{max} is approximately 44.

Algorithm 1 is not only theoretically efficient, but also fast in practice. We have implemented the algorithm and tested its efficiency on our data. For instance, processing a multigraph of 3.3 million edges takes less than 10 seconds on a desktop PC.

4.3 Computing TSG probabilities

The algorithm in the previous section produces a set of TSGs for each past time window. Based on that information, we want to be able to estimate the likelihood of any new TSG. One way to address that is to create a database of all encountered TSGs and, given any new TSG T, to determine whether a TSG identical to T is contained in the database. The problem with such an approach is that, because of the huge number of possible TSGs, it will happen very often that T is not in the database, generating many false positives. Moreover, the result of any query is binary, i.e., T is either in the database or not, and there is nothing in-between to allow a more varied and nuanced assessment.

For that reason, we develop a user model that allows, for any TSG T of the protocol graph G, to compute a number between 0 and 1 that corresponds to the probability that T occurs in the network traffic based on the available data. We denote by S the set of all TSGs encountered so far or encountered in a past time window of a specified time length. We define, for each node v of G, a number $\text{wt}(v)$ equal to the number of all edges in G with source v. For each edge e of G we define a number $\text{wt}(e)$ equal to the number of all parallel edges in G with the same endpoints as e. The user model we design assumes that the TSGs are generated as random graphs using the following process:

(i) Choose a random size s of a TSG based on the frequency of the sizes in S;

(ii) Choose a random start node r among all nodes of G with probability proportional to $\text{wt}(r)$ and add it to T;

(iii) Until the size (the number of edges) of T is s, add an edge e to T among all edges whose sources are nodes already in T and with probability proportional to $\text{wt}(e)$.

This model directly leads to an algorithm for assigning a probability to any TSG T, defined as the product of all probabilities from steps (i), (ii) and all steps (iii) in the user-model procedure. For computing the individual probabilities we need to compute and dynamically maintain the values $\text{wt}(v)$ and $\text{wt}(e)$ defines above, the numbers $\text{sz}(s)$ of all TSGs of a given size s, the number $\text{wt}(G)$ of all edges of G, and the number of all edges with sources nodes currently in T. For instance, if T has a root node r, the probability from step (i) is $\text{sz}(|T|)/|S|$ and the probability from step (ii) is $\text{wt}(r)/\text{wt}(G)$.

5. STATISTICAL ANALYSIS METHODOLOGY

5.1 Approach overview

Given traffic log data, we divide the entire time interval covered by the data into 10 minute subintervals we call *time windows*. Then, for each time window, we compute a TSG decomposition and corresponding probability scores. The goal is to monitor the TSGs over time and flag subgraphs that are anomalous in the sense of having a combination of unusually large size and a rare set of edges. Without having information on usage patterns, an intruder will find it difficult to avoid creating such subgraphs.

Let n_t denote the number of TSGs in the decomposition for time window t. For the i-th TSG, let $x_{i,t}$ denote its size (number of edges) and $y_{i,t}$ denote its probability score. The following subsections describe a method for modeling TSG decompositions in terms of the quantities n_t, $x_{i,t}$ and $y_{i,t}$.

The main idea of our approach is that, in the training module, we represent $x_{i,t}$ and $y_{i,t}$ as random variables coming

from a probability distribution that depends on a number of parameters which are functions of t. By substituting $x_{i,t}$ and $y_{i,t}$ with the values of the observations in the given time window, we obtain a system of equations that can be used to estimate the parameters. Based on that information, we get a predictive model for future values of $x_{i,t}$ and $y_{i,t}$. In the detection module, if the values we actually observe deviate substantially from the prediction, we trigger an anomaly alert.

The main challenges in implementing this idea are (i) the distribution of the probability score $y_{i,t}$, even after conditioning on the value of the size $x_{i,t}$, is very heavily tailed under normal network activity, (ii) the distribution of the size $x_{i,t}$ is not modeled well by typical parametric models, and (iii) the distributions of $x_{i,t}$ and $y_{i,t}$ are likely to exhibit diurnal trends. In the following we describe a novel approach to modeling $x_{i,t}$ and $y_{i,t}$, which explicitly addresses all of these issues.

5.2 Modeling the Data

We model the probability scores $y_{i,t}$ conditional on a given size x as coming from a *noncentral t-distribution* [15] with a mean that varies over x and all parameters varying smoothly over time t. The noncentral t is used to capture skewness and heavy-tailed behavior that is observed in plots similar to the probability scores calculated from our data set. Using a family of distributions that includes heavy tails can better accommodate the diversity of probability scores and lead to more controllable false alarm rates when monitoring for anomalies.

The sizes of the TSGs $x_{i,t}$ are modeled by a flexible failure time type distribution, called the *discrete hazard* (DH) model [9], with a discrete hazard rate that varies smoothly over time. We use the DH model in order to avoid specifying a specific parametric form for the distribution of size, which may be somewhat restrictive. The DH allows for *any* discrete distribution, while encouraging smoothness of the hazard rate is an intuitive way to limit the degrees of freedom involved in the estimation.

We will use the logit function defined as $\text{logit}(y) = \log[y/(1-y)]$ in order to transform the TSG probability scores from the interval $(0, 1)$ to the real line for numerical stability and convenience of modeling with the noncentral t-distribution. Moreover, recall that $\overset{ind}{\sim}$ denotes "independently distributed as" and $\overset{iid}{\sim}$ denotes "independently and identically distributed as".

In our model, at each time $t \in \{1, 2, \dots\}$, we have observations $(x_{i,t}, y_{i,t})$, $i = 1, \dots, n_t$, where the size $x_{i,t}$ takes on positive integer values, and the probability score $y_{i,t}$ is in the interval $[0, 1]$. We model the observations as

$$\text{logit}(y_{i,t}) \mid x_{i,t} \overset{ind}{\sim} \mathcal{T}(\mu_t(x_{i,t}), \sigma_t^2, \nu_t, \eta_t) \quad (2)$$

$$x_{i,t} \overset{iid}{\sim} \text{DH}(\pi_t), \quad (3)$$

where $\mathcal{T}(m, v, \nu, \eta)$ represents a scaled noncentral t distribution with mean m, variance v, degrees of freedom ν, and noncentrality parameter η and $\text{DH}(\pi)$ is a distribution with discrete hazard (DH) function $\pi \equiv \pi(x)$:

$$\Pr(X = x|\pi) = \pi(x) \prod_{j=1}^{x-1} (1 - \pi(j)), \text{ for } x = 1, 2, \dots. \quad (4)$$

As a result of the representations (2), (3), and (4), $x_{i,t}$ and $y_{i,t}$ are defined as functions based on parameters $\mu_t, \sigma_t^2, \nu_t, \eta_t$, and π_t. The functions μ and π each depend on the TSG size x, and we parameterize them in terms of basis functions:

$$\mu_t(x) = \sum_{j=1}^{m_\mu} \beta_{t,j}(x) A_j(x) \quad (5)$$

$$\text{logit}[\pi_t(x)] = \sum_{j=1}^{m_\pi} \gamma_{t,j}(x) B_j(x), \quad (6)$$

The functions $A_1, \dots, A_{m_\mu}$ and $B_1, \dots, B_{m_\pi}$, are basis functions that need to be specified. Based on visual inspection of the data (e.g., Figures 4 and 5) we choose both $\{A_j\}$ to be linear spline bases with knots at $x = \{2, 3, 4, 5, 6, 8, 10, 12, 14, 16, 18, 20, 25\}$. We also take $\{B_j\}$ to be a *natural* linear spline basis functions with knots at $x = \{2, 3, 4, 5, 6, 8, 10\}$. The natural linear spline for $\pi(x)$ is chosen in part because we need to ensure that the hazard function is regular, in the sense that $\Pr(X < \infty) = 1$ in (4). This property is guaranteed if $\pi(x)$ is constant for large x, as is the case for a natural linear spline. We chose to place the largest knot at $x = 10$ for $\pi(x)$ to guarantee enough data past the last knot for purposes of stability in estimation. Estimation of the mean function $\mu(x)$ is much more stable and could thus tolerate more knots.

Therefore, the x variables depend on parameters $\boldsymbol{\theta}_t = [\beta_{t,1}, \dots, \beta_{t,m_\mu}, \sigma_t^2, \nu_t, \eta_t]$ and the y variables depend on parameters $\boldsymbol{\gamma}_t = [\gamma_{t,1}, \dots, \gamma_{t,m_\pi}]$. Estimation of the parameters $\boldsymbol{\theta}_t$ and $\boldsymbol{\gamma}_t$ is conducted via a regularized likelihood based approach detailed in the Appendix. Specifically, we maximize a penalized likelihood, which is exponentially weighted back in time in a manner to explicitly account for diurnal patterns in the data.

As a result, we compute the one-step-ahead estimates for $\boldsymbol{\theta}_t$ and $\boldsymbol{\gamma}_t$, which we denote by $\hat{\boldsymbol{\theta}}_{t|t-1}$ and $\hat{\boldsymbol{\gamma}}_{t|t-1}$, respectively.

5.3 Using the Model to Detect Anomalies

We suppose that a given pair $(x_{i,t}, y_{i,t})$ demonstrates evidence of an anomaly in this setting if $x_{i,t}$ is *large* and $y_{i,t}$ is *small*. This is a TSG with a large size and a small probability score. Let X_t and Y_t represent random variables from the model (2) and (3) with one-step predictive parameters $\hat{\boldsymbol{\theta}}_{t|t-1}, \hat{\boldsymbol{\gamma}}_{t|t-1}$. For a given time t, we calculate the probabilities of exceeding the observed values, $(x_{i,t}, y_{i,t})$ $i = 1, \dots, n_t$, then combine them together in a rigorous manner to create a single p-value for anomalousness at time t. Specifically, at time t, for each $(x_{i,t}, y_{i,t})$, we calculate

$$\rho_{i,t,1} = \Pr(X_t > x_{i,t} \mid \hat{\pi}_{t|t-1}) = \prod_{x-1}^{x_{it}} \left(1 - \hat{\pi}_{t|t-1}(x)\right)$$

$$\rho_{i,t,2} = \Pr(Y_t > y_{i,t} \mid \hat{\mu}_{t|t-1}, \hat{\sigma}^2_{t|t-1}, x_{it}) \quad (7)$$

$$= \Phi(y_{it}; \hat{\mu}_{t|t-1}, \hat{\sigma}^2_{t|t-1}).$$

where $\Phi(\cdot; m, v)$ is the cumulative distribution function of a normal distribution with mean m and variance v. We then use

$$\omega_t = \min_{i=1,\ldots,n_t} \{\rho_{i,t,1} \cdot \rho_{i,t,2}\}, \quad (8)$$

to measure *anomalousness* of the TSGs in the t-th time bin. Assume that ω_t represent a random variable with a distribution equal to that of the minimum product of n_t pairs of uniform variables (under the assumption that network activity follows our model at time t, the ρ's will have approximately uniform distributions). This means that the distribution of ω_t is that of the minimum of n_t independent observations from negative-log-Gamma(2,1) distribution. The the p-value for anomalousness at time t is then

$$p_t = \Pr(\Omega_t < \omega_t) = 1 - [\mathcal{G}(-\log \omega_t; 2, 1)]^{n_t}, \quad (9)$$

where $\mathcal{G}(\cdot; a, b)$ is the cumulative distribution function (CDF) of the Gamma distribution with shape and rate parameters a and b, respectively. We set a threshold c (e.g., c=0.001) and if $p_t < c$ at some time then the identified TSG (the i that minimizes (8)) can be sent to an analyst for further inspection. In practice, c will need to be set to a level to produce a tolerable number of false alarms per week.

6. EXPERIMENTS

For our experiments we have used a subset of the LANL traffic data set described above covering a two-month period and, for each 10 minute time window, we have computed a TSG decomposition and corresponding probability scores. Figure 3 displays a time series of $Z_t = -\Phi(p_t)$ over one month of data, where Φ is the standard normal CDF, so that large values of Z_t indicate anomaly on a normal scale. The most significant anomaly occurs on Tuesday of the first week. The data point corresponding the this time bin is highlighted in red (point A).

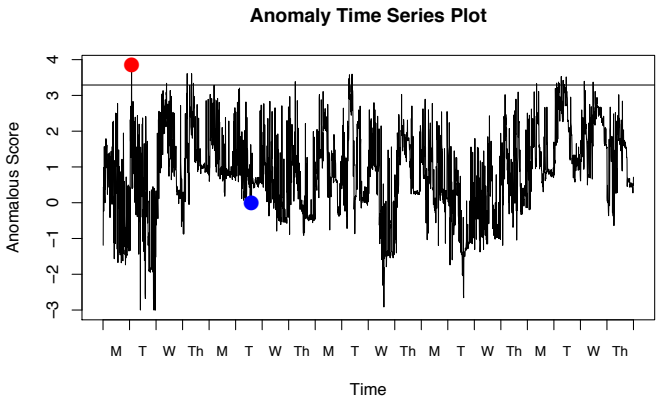

Figure 3: Anomaly time series plot over four weeks of SSH data from LANL's internal network during August 2009.

Figure 4 shows a one-step-ahead predictive model for $\boldsymbol{y}_t \mid \boldsymbol{x}_t$ during the time bin of the most significant anomaly with the data point A. To contrast this, Figure 5 displays the same plot during a more typical time bin which corresponds to the data point highlighted in blue in Figure 3 (point B). The data point B corresponds to very "typical" TSG. The most significant anomaly, point A, has a large TSG probability score $y_{i,t}$ along with a large size, which both contributed to the small p_t value.

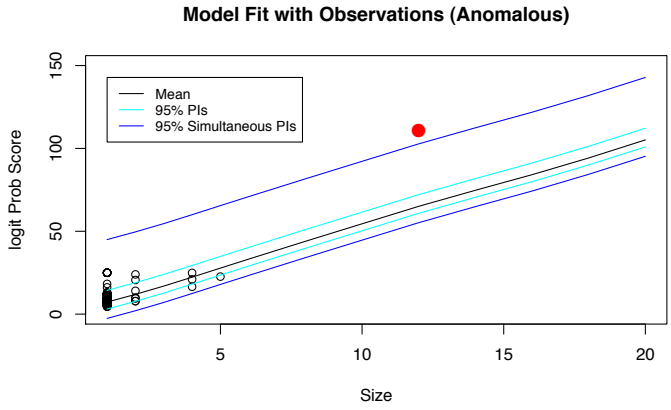

Figure 4: One-step-ahead fitted probability score of TSG versus size during the anomalous time bin.

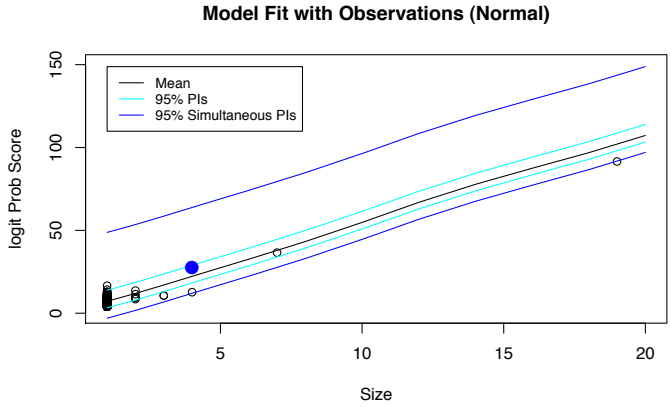

Figure 5: One-step-ahead fitted probability score of TSG versus size during a typical time bin.

Figure 6 shows a one-step-ahead complementary cumulative distribution function (CCDF) (i.e., one minus the CDF) for $\boldsymbol{x}_t$ using the one-step-ahead cross-validated (CV) [13] optimal tuning parameters to fit the model along with the actual data for a given time point. It shows the probability of exceeding a given size decreases rapidly at first, then flattens out behaving like a geometric distribution in the tail (linear on a log scale as in the figure).

Figure 7 displays the anomalous subgraph with heat colored edges to indicate the relative contribution of each edge to the anomaly score. This plot can be used as a diagnostic tool, and passed to a computer security specialist for aid in further analysis. For example, it can be inferred from the plot that the edges $(1, 7)$, $(1, 8)$, $(1, 9)$, are the most responsible for the anomaly and as such should be the starting point for further investigation.

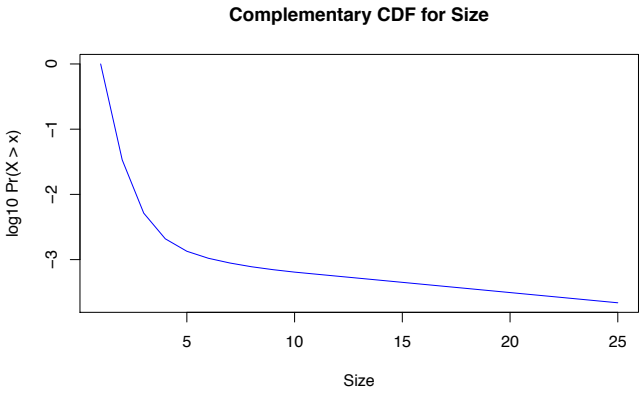

Figure 6: One-step-ahead fitted *complementary* CDF for Size during the anomalous time bin on a logit scale. The probability of a size of $X > 12$ (i.e., the size of the TSG responsible for the anomaly) is 5.37×10^{-4}.

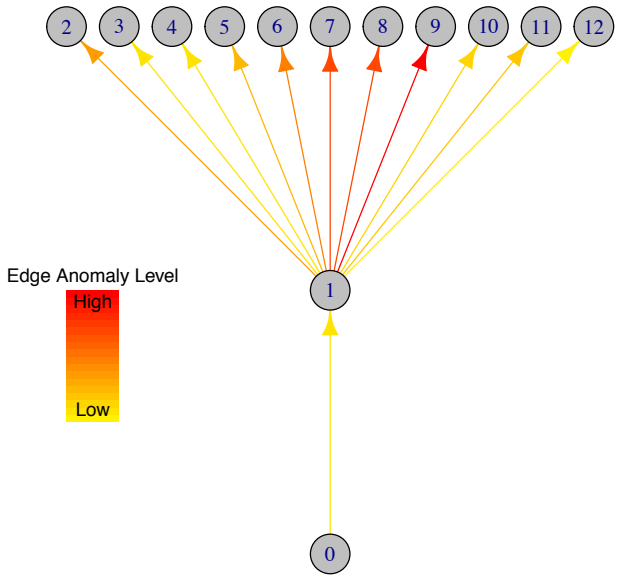

Figure 7: The anomalous subgraph leading to the red data point in Figure 3 with heat colored edges to indicate the edge contribution to the anomaly

7. CONCLUSION

We have described a methodology for representing the SSH communications in a large network as a time-labeled graph and decomposing that graph into subgraphs corresponding to interrelated sessions we call telescoping subgraphs (TSGs). We also described a statistical approach for analyzing TSG decompositions that takes into account the diurnal patterns of the communications and computes on that basis a predictive model for future traffic that can be used to detect anomalies. Our research leaves a number of problems for future study, such as analyzing the dependency of the results on the tie-breaker method used when defining TSGs, e.g., latest-first vs. earliest-first, developing other models that can be used to compute the probability of a TSG, and optimizing the model parameters. It will also be interesting to check how the approach works on other network protocols.

8. ACKNOWLEDGMENTS

The authors acknowledge and appreciate the support provided for this work by the Los Alamos National Laboratory Directed Research and Development Program (LDRD, project number 20110093DR). They would like also to thank Aric Hagberg for many helpful discussion and for drawing two of the figures.

9. REFERENCES

[1] M. Collins. *A Protocol Graph Based Anomaly Detection System*. Ph.D. Thesis, Carnegie Mellon University, Pittsburgh, PA 15213, April 2008.

[2] M. P. Collins and M. K. Reiter. Hit-list worm detection and bot identification in large networks using protocol graphs.

[3] J. Glaz, J. Naus, and S. Wallenstein. *Scan Statistics*. Springer-Verlag, 2001.

[4] N. Heard, D. Weston, K. Platanioti, and D. Hand. Bayesian anomaly detection methods for social networks. *Annals of Applied Statistics*, 4(2):645–622, 2010.

[5] M. Iliofotou, M. Faloutsos, and M. Mitzenmacher. Exploiting dynamicity in graph-based traffic analysis: Techniques and applications. In *ACM CoNEXT*, New York, NY, USA, December 2009. ACM.

[6] M. Iliofotou, P. Pappu, M. Faloutsos, M. Mitzenmacher, S. Singh, and G. Varghese. Network traffic analysis using traffic dispersion graphs (TDGs): techniques and hardware implementation. Technical report, UCR, May 2007.

[7] M. Kulldorff. A spatial scan statistic. *Communications in Statistics- Theory and Methods*, 26(6):1481–1496, 1997.

[8] D. Lambert and C. Liu. Adaptive thresholds: Monitoring streams of network counts. *Journal of the American Statisitcal Association*, 101:78–88, 2006.

[9] N. Mantel and B. Hankey. A logistic regression analysis of response-time data where the hazard function is time dependent. *Communications in Statistics - Theory and Methods*, 7(4):333–347, 1978.

[10] J. Naus. Approximations for distributions of scan statistics. *Journal of the American Statistical Association*, 77(377):177–183, 1982.

[11] J. Naus and W. S. Temporal surveillance using scan statistics. *Statistics in Medicine*, 25(2):311–324, 2006.

[12] C. C. Noble and D. J. Cook. Graph-based anomaly detection. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '03, pages 631–636, New York, NY, USA, 2003. ACM.

[13] R. R. Picard and R. D. Cook. Cross-validation of regression models. *Journal of the American Statistical Association*, 79(387):pp. 575–583, 1984.

[14] C. E. Priebe, J. M. Conroy, and D. J. Marchette. Scan statistics on enron graphs. *In Workshop on Link Analysis, Counterterrorism and Security, Newport Beach, CA*, 2005.

[15] A. van Aubel and W. Gawronski. Analytic properties of noncentral distributions. *Applied Mathematics and Computation*, 141(1):3–12, 2003.

APPENDIX

A. MODEL ESTIMATION DETAILS

Here we describe the estimation procedure for the parameters of the model in Section 5.2. We use a penalized likelihood, which is exponentially weighted back in time in a manner to explicitly account for diurnal patterns in the data.

Let $\boldsymbol{y}_t = (y_{1,t}, \ldots, y_{n_t,t})'$ and $\boldsymbol{x}_t = (x_{1,t}, \ldots, x_{n_t,t})'$. The log-likelihood for $\boldsymbol{y}_t \mid \boldsymbol{x}_t$ is given by

$$l(\boldsymbol{y}_t; \boldsymbol{\theta}_t, \boldsymbol{x}_t) = \sum_{i=1}^{n_t} \log \phi\left(\text{logit}(y_{i,t}); \mu(x_{i,t}), \sigma^2, \nu_t, \eta_t, \right) \quad (10)$$

where $\phi(\cdot; m, v, \nu, \eta)$ is the density function for a scaled noncentral t distribution with mean m and variance v, degrees of freedom ν, and noncentrality parameter η. Further, let $\boldsymbol{y}_{1:t} = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{t-1}, \boldsymbol{y}_t]$, and $\boldsymbol{x}_{1:t} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{t-1}, \boldsymbol{x}_t]$. The weighted log-likelihood for $\boldsymbol{y}_{1:t}$ is given by

$$l(\boldsymbol{y}_{1:t}; \boldsymbol{\theta}_t, \boldsymbol{x}_{1:t}, \boldsymbol{h}_y) = \sum_{s=1}^{t} \left\{ \left[\sum_{d=0}^{\infty} K(t, s, d, \boldsymbol{h}_y) \right] l(\boldsymbol{y}_s; \boldsymbol{\theta}_t, \boldsymbol{x}_s) \right\}, \quad (11)$$

and $K(t, s, d, \boldsymbol{h}_y)$ is a weight function that decomposes as

$$K(t, s, d, \boldsymbol{h}_y) = K_D(d, h_{y,d}) K_B(t, s, d, h_{y,b}) \quad (12)$$

where K_D is a daily component and K_B is a bin-to-bin component described below and where $\boldsymbol{h}_y = (h_{y,d}, h_{y,b})$ denote tuning parameters to be chosen via cross-validation as discussed below. The individual kernels that make up K are,

$$K_D(d, h) = (1 - h)^d, \quad (13)$$

$$K_B(t, s, d, h) = (1 - h)^{|(t - d*T) - s|}, \quad (14)$$

where T is the number of time bins per day. This is similar in nature to the EWMA approach for data with diurnal trends discussed in [8]. Figure 8 displays the intuitive nature of the weight function K, using half-hour time bins to aid illustration, instead of the 10-minute bins used for the data analysis.

Finally, one-step-ahead estimates $\hat{\boldsymbol{\theta}}_{t|t-1}$ for $\boldsymbol{\theta}_t$ are obtained by maximizing the expression

$$-\, l(\boldsymbol{y}_{1:t-1}; \boldsymbol{\theta}_t, \boldsymbol{x}_{1:t-1}, \boldsymbol{h}_y) + \lambda_\mu \sum_{x=3}^{\infty} [D^2 \mu(x)]^2 \quad (15)$$

over $\boldsymbol{\theta}$, where

$$D^2 g(x) = g(x) - 2g(x-1) + g(x-2)$$

is the second difference penalty operator. The penalty represents the sum of the squared second order differences for $\mu(x)$, which is a discrete analog of the common practice of penalizing the integral of the squared second derivative. This penalties encourage smooth mean function. For our parametrization using linear splines, the penalty is equivalent to the sum of the squared changes in slopes at the knots. The penalty weight λ_μ is selected by cross-validation as described in more detail below.

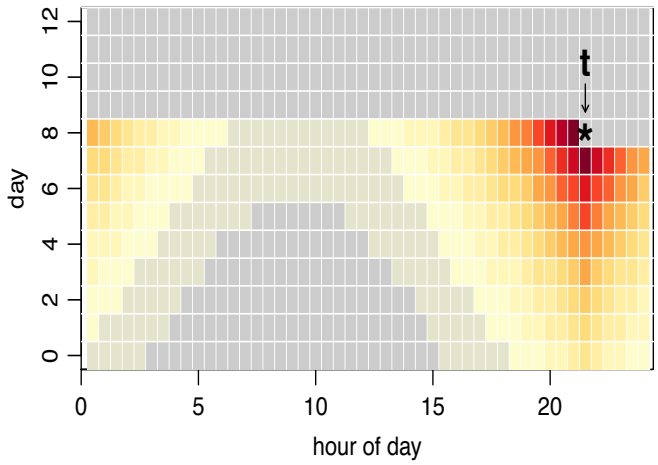

Figure 8: Geometric display of the weight function K. This example uses half-hour bins and the smoothing parameters are $h_{y,d} = 0.17$ and $h_{y,b} = 0.13$, corresponding to effective bandwidths of 11 days and 14 half-hour bins respectively.

The coefficients $\boldsymbol{\gamma}_t$ for the hazard function $\pi(x)$, are estimated in a similar fashion, i.e., with a kernel weight function over time bin and between consecutive days and a $D^2 \text{logit} \pi(x)$ penalty to encourage smoothness in the hazard rate function over x.

The two sets of tuning parameters for $\boldsymbol{y}$ and $\boldsymbol{x}$, respectively, need to be chosen via cross-validation. This is done, by computing the predictive (out of sample) likelihood for $\boldsymbol{y}_t$ using the one-step-ahead predictive models for $\boldsymbol{y}_t \mid \boldsymbol{x}_t$and $\boldsymbol{x}_t$, respectively.